Enqpy™ Stream Cipher

Formal Cryptographic Description

Complete Specification of the OWC, PDAF, and PDAF_SEC Primitives

Including Security Profiles, Key Management, and NIST SP 800-22 Validation

Revision 1.0 — 2026

NQP LLC • www.enqpy.com • Public

Paul McGough

NQP LLC

Manassas, Virginia • RPM@enqpy.com

Copyright © 2026 Paul McGough / NQP LLC. Released for evaluation, citation, and academic use; the Enqpy™ patents are bound open under an irrevocable public covenant that leaves the cipher free to use, port, and deploy at any scale, including commercially (see the LICENSE file in the Enqpy reference repository for full terms). This document is the public Formal Cryptographic Description (FCD) of the Enqpy™ cipher; commercial implementation optimization techniques are documented separately in the Enqpy™ Implementation Companion (NQP LLC; available to commercial licensees, Partner Program participants, and Foundation Conformance Program reviewers under applicable separate agreement). Enqpy™ is a trademark of the Enqpy™ Foundation Inc., licensed for commercial use by NQP LLC under the Foundation–NQP IP License Agreement.

Document Overview

This document is the complete formal specification of the Enqpy™ stream cipher as developed by NQP LLC. It defines the Ideal Enqpy™ Configuration — the single proved-secure implementation — along with the OWC (One-Way Computation) and PDAF (Pseudo-random Data Augmentation Function) cryptographic primitives, the PDAF_SEC encrypt/decrypt function, key management architecture, security analysis, and NIST SP 800-22 statistical randomness validation results. EnqpyADS™ Built-To-Application variant configurations — including HMIX-based and other arrangements suited to specific deployment requirements — are described in the EnqpyADS™ specification.

Enqpy™ claims security via mathematical underdetermination rather than computational hardness, satisfying Shannon's Ideal System definition with a finite key.

Formal Proof Status — First Provably Secure Shannon Ideal System

As of 2026, Enqpy™ is the first finite-key cipher construction to carry a formal information-theoretic proof of the Shannon Ideal System property — a specific information-theoretic property distinct from computational security. The One-Time Pad achieves Perfect Secrecy (a strictly stronger property) but requires a key equal in length to the message, making it impractical for general use. Enqpy™ achieves the Ideal System property — non-vanishing key equivocation under unlimited ciphertext accumulation — with a fixed, reusable finite key, in the Ideal Enqpy Configuration defined in companion document [12].

This claim is formally established in companion document [12]:

- [12] Enqpy™ Stream Cipher — Formal Information-Theoretic Proof (Rev 1.0, 2026). Establishes Lemma 1 (MOD16 fiber size), Theorem 1 (PDAF Mode 1 preimage lower bound $|P(O^*)| \geq 2$ worst case / ≥ 16 non-degenerate), Theorem 2 (Shannon Ideal System — proved for Ideal Enqpy Configuration: $H(EK,QK|T^{\wedge\infty}) = \log_2(4) = 2$ bits exactly; exact conditional independence $T_{>t} \perp (EK, QK) \mid T_{\leq t}$), Theorem 3 (plaintext equivocation: $H(PT|CT,OR) \geq 128$ bits unconditionally for HIGH profile), Corollary 2 (quantum invariance). §11 (Appendix B) provides the extended algebraic proofs: exact preimage count formula, equivocation tightness over $\mathbb{Z}_{16}$, cycle case completeness, and exact solution count with cycles. The [+8] global shift invariant (Remark 6.1) extends to a full ciphertext-equivalent coset under the key role separation of the Ideal Configuration (see [12] §6.1). Also defines the Ideal Enqpy Configuration and formally characterizes the standard deployment profiles (LOW/MED/HIGH) and their security trade-offs.

All proofs are unconditional: no computational hardness assumption is invoked. The bounds hold against any adversary, including quantum adversaries.

This document presents those claims, their formal basis, operational constraints, and the complete public implementation specification, including the C reference implementation, API, test vectors, and public performance record. Commercial optimization basis and implementation-engineering techniques are documented separately in the Enqpy™ Implementation Companion.

Ideal Enqpy™ Configuration

The Ideal Enqpy™ Configuration is the specific key derivation arrangement under which the construction is formally proved to satisfy Shannon's Ideal System definition — $H(EK, QK \mid T^{\wedge\infty}) = \log_2(4) = 2$ bits exactly, unconditional and quantum-invariant. It is defined by the Key Role Separation

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Principle: each master key appears as the ValueKey parameter in exactly one PDAF Mode 1 call per session; neither master key appears as an OffsetKey parameter or HMIX input.

OffsetKey parameters (VKP, OKP) are derived from the public nonce OR_t only. Operational requirements are specified in §8.9. Deployments requiring the formal Shannon Ideal System guarantee SHOULD use this configuration. EnqpyADS™ HMIX-based variant configurations may use the same LOW/MEDIUM/HIGH key-size profiles for deployment sizing, but they are variants and do not carry the formal global Shannon Ideal System guarantee unless separately proved.

NOTE TO EDITORS

This document describes a proprietary cryptographic system whose terminology, notation, and security claims are defined precisely within the text. Editors are asked to preserve all technical language exactly as written — including defined terms, symbolic notation, and capitalization conventions (e.g., OWC, PDAF, PDAF_SEC, OR, VK, OK, CS) — as these carry specific formal meaning and any alteration may introduce technical inaccuracy. Security claim language in particular should not be softened, qualified, or paraphrased without author consultation. Questions regarding technical content should be directed to the author rather than resolved editorially.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

1. Introduction

1.1 Scope

Enqpy™ (pronounced “En-Q-P” or “N-Q-P”) is a symmetric-key stream cipher developed by NQP LLC. Unlike all commercially deployed stream and block ciphers, which derive their security from assumed computational hardness — the difficulty of factoring large integers or solving discrete logarithm problems — Enqpy™ claims security via mathematical underdetermination: the ciphertext is genuinely consistent with multiple plaintexts under any key search, and no additional computation can resolve the ambiguity because the required information is structurally absent.

Enqpy™ has been formally proved to satisfy Claude Shannon’s Ideal System definition from his 1949 paper “Communication Theory of Secrecy Systems” — the finite-key variant in which the equivocation $H(EK, QK \mid C^m, M^m)$ does not approach zero as the amount of intercepted material $m \to \infty$. See [12].

This document specifies the Ideal Enqpy™ Configuration as the primary and formally proved-secure implementation. It covers: the OWC (One-Way Computation) primitive; the PDAF (Pseudo-random Data Augmentation Function) in two modes; the PDAF_SEC encrypt/decrypt function in its Ideal Configuration; key size requirements; key derivation, session key generation, and in-cipher key update; Nil-Communication Key Update; security analysis; NIST SP 800-22 randomness test results; and C and VHDL reference implementations with test vectors. EnqpyADS™ variant configurations for other deployment requirements are documented separately.

1.2 Shannon’s Ideal System — Background

Shannon proved that a cryptosystem achieves Perfect Secrecy if and only if: (1) the key space is at least as large as the message space, and (2) every key is equally likely. The One-Time Pad satisfies this but requires a key as long as the message, rendering it impractical for most applications.

Shannon also described a weaker but still strong notion: an Ideal System, defined as a cipher in which “no matter how much material is intercepted, the enemy still does not obtain a unique solution to the cipher but is left with many alternatives, all of reasonable probability.” This is formally expressed as the equivocation $H(EK, QK \mid C^m, M^m)$ not approaching zero as $m \to \infty$.

Enqpy™ has been proved to realize this Ideal System property for arbitrary binary data using a finite key, by means of the underdetermined PDAF key expansion function applied at every step of key derivation, selection, and update. The underdetermination arises from the many-to-one property of MOD16 modular arithmetic: for any output nibble r, exactly 16 input pairs (a, b) satisfy $(a + b) \bmod 16 = r$. This structural property is independent of computational hardness and is preserved regardless of adversary computing power, including quantum computers. The formal proof appears in companion document [12].

The Ideal System claim applies specifically to the cipher’s key equivocation. Applications requiring plaintext confidentiality AND ciphertext integrity must apply the MAC and integrity guidance provided in Section 8.6.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

2. Notation and Conventions

Symbol	Definition
EK	Encryption Key — master secret, shared out-of-band. MUST be independently generated from QK. EK ≠ QK required. Minimum length per security profile (Section 3).
QK	Query Key — companion master secret, shared out-of-band. MUST be independently generated from EK.
OR	Open Return — per-message public random nonce. MUST be generated by a CSPRNG, used exactly once per (EK, QK) credential pair, and never reused. Transmitted in clear alongside the ciphertext.
OR_CTR	Open Return Counter — a 64-bit monotonically increasing integer bound to each (EK, QK) credential pair. Incremented before each OR generation.
OR_EXP	OR Counter Expansion — a PDAF-based expansion of OR_CTR to n nibbles, used in Phase 1 mixing.
eff_or	Effective Open Return — OR value used in all key derivation steps, computed by mixing the raw OR with OR_EXP per Phase 1. Transmitted in place of raw OR.
VKP	ValueKey Pointer = $PDAF_1$ (eff_or, eff_or)[:n] — per-session pointer key (Ideal Configuration; nonce-derived, public).
OKP	OffsetKey Pointer = (VKP + DS_SEP) mod 16 — per-session offset key (Ideal Configuration; domain-separated from VKP, public).
VKC	ValueKey Cipher = PDAF(EK, VKP, Mode=1) — per-session working value key.
OKC	OffsetKey Cipher = PDAF(QK, OKP, Mode=1) — per-session working offset key.
CS	Case Selector — per-session, key-derived 3-nibble value that determines the Selection case ordering. Derived from VKC and OKC using PDAF Mode 1.
W	Per-unit message key (cipher keystream unit), output of the CS-ordered Selection phase.
n	Key length in 4-bit hex nibbles. Minimum: 32 nibbles (128 bits, LOW profile); standard: 64 nibbles (256 bits, HIGH profile).
n^2	Full PDAF output length (n × n cross-product nibbles).
⊕	Bitwise XOR.
HMIX(a,b,d)	Nibble-wise MOD16 combination: HMIX[i] = MOD16_TABLE[a[i]][MOD16_TABLE[b[i]][d]].
DS_SEP	Ideal Configuration domain separator constant for OKP derivation. Required value: 0xF. OKP[i] = (VKP[i] + DS_SEP) mod 16.
DS_VK	Domain separator 0x5 used in EnqpyADS™ HMIX-based variant configurations. Not used in the Ideal Configuration.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Symbol	Definition
DS_OK	Domain separator 0xA for HMIX-based OKP derivation in EnqpyADS™ variant configurations. Not used in the Ideal Configuration (which uses DS_SEP = 0xF).
MOD16_TABLE	16×16 lookup table: MOD16_TABLE[a][b] = (a + b) & 0xF.

Hexadecimal strings are uppercase. Key lengths are byte-aligned (even number of hex digits). The fundamental unit in all PDAF/OWC operations is the 4-bit nibble (hexadecimal digit, value 0–15), stored as a uint8_t.

3. Security Profiles and Minimum Requirements

3.1 Overview

Enqpy™ is built on the Ideal Configuration defined in §8.9, which is the single formally proved-secure implementation. The Ideal Configuration supports flexible key sizes meeting the minimum requirements below. For deployments with specific security/performance/implementation constraints — including defense-in-depth key mixing, FPGA optimization, IoT resource constraints, or other EnqpyADS™ Built-To-Application requirements — variant configurations are available through the EnqpyADS™ specification. Variant configurations below the key size minimums below are prohibited.

3.2 Security Profile Definitions

Profile	Min Key Size	Min OR Entropy	PDAF Output	Recommended Use
LOW	128 bits (32 nibbles)	128-bit	n^2 = 1,024 nibbles	IoT, low-power embedded; noncritical data. Ideal Configuration at n=32. Note: provides a reduced quantum security margin — see Section 3.5.
MEDIUM	192 bits (48 nibbles)	192-bit	n^2 = 2,304 nibbles	Enterprise communications, cloud services.
HIGH	256 bits (64 nibbles)	256-bit	n^2 = 4,096 nibbles	Government, defense, critical infrastructure. Default recommendation for all new deployments.

The HIGH profile (256-bit / 64-nibble keys) is the default recommendation for all new deployments. The LOW and MEDIUM profiles are available for hardware-constrained environments and require explicit documented justification.

3.3 Prohibited Configurations

© 2026 NQP LLC • All rights reserved • www.enqpy.com

The following configurations are prohibited under all security profiles: key length n < 32 nibbles (128 bits); EK = QK (identical master keys); EK or QK with zero entropy (all-zero, all-same-value, or repeating-pattern keys); OR reuse with the same (EK, QK) credential pair under any circumstances; OR generated by a non-CSPRNG source; and key size reductions below the active security profile minimum during operation.

EK ≠ QK — ALGEBRAIC RATIONALE AND ENFORCEMENT

Algebraic analysis: in the Ideal Configuration, VKP is derived from eff_or and OKP is derived as (VKP + DS_SEP) mod 16, so the two pointer paths remain distinct even when EK = QK. The prohibition on EK = QK is therefore not because the cipher fails to operate; it is because identical master keys collapse the intended two-key architectural separation. With EK = QK, recovery of either master key via UP-direction attack yields both master keys simultaneously, eliminating the two-key architectural separation. With EK ≠ QK, a breach of one master key does not expose the other. The prohibition preserves the independence of the two-key system. Enforcement: PDAF_SEC implementations SHOULD return −1 with an appropriate error code if EK and QK are byte-identical. Callers MUST verify EK ≠ QK at credential generation time and before each Nil-Communication Key Update. The cipher-layer check is a defense-in-depth measure; application-layer enforcement at credential generation time is the primary control.

SINGLE-BARRIER ARCHITECTURE — REQUIRED DISCLOSURE

The PDAF Mode 1 gate is the sole cryptographic barrier protecting EK and QK. HMIX is a structural domain separation mechanism, not a cryptographic barrier — it is algebraically invertible given either input and the domain separator constant. Any reduction in the Mode 1 preimage count below its proved minimum directly reduces the master key security margin by the same factor. The architecture is a single-barrier design; this is correct by intent. The HIGH profile preimage lower bound is ≥ 2 (worst case: μ_odd ≥ 1) or ≥ 16 (non-degenerate: μ_odd = 0) per compatible OffsetKey, unconditional and holding against any adversary including quantum adversaries (proved in [12], Theorem 1 and Corollary 2). Deployers must understand that PDAF Mode 1 non-invertibility is the exclusive cryptographic foundation of master key protection. No secondary barrier exists and none is required given the formal proof bounds. Independent peer review of the preimage lower-bound proof ([12], Theorem 1) is the highest-priority assurance action for evaluators of this architecture; see Section 8.8 for proof status.

3.4 Key Entropy Requirements

All master keys (EK, QK) MUST be generated by a Cryptographically Secure Pseudo-Random Number Generator (CSPRNG) meeting the requirements of NIST SP 800-90A, SP 800-90B, or equivalent. EK and QK MUST be generated independently from separate CSPRNG invocations. The min-entropy of EK and QK MUST be ≥ n/2 nibbles (≥ 128 bits for n = 64). The OR MUST be generated per-message by a CSPRNG with entropy meeting the profile minimum.

The OR_CTR MUST increment monotonically before each use and MUST persist across power cycles for each (EK, QK) credential pair. For HIGH profile deployments, hardware-secured monotonic counter storage is required to prevent counter state loss from power interruption.

KEY ENTROPY ADVISORY

The n/2-nibble minimum entropy floor is the absolute minimum for conformance. Keys generated at this minimum provide quantum search resistance of 2^(H_min/2), where H_min is the actual min-entropy in bits. For HIGH profile keys at the minimum entropy floor (128-bit min-entropy), quantum search cost is 2^{64} — equivalent to the LOW profile quantum margin, not the HIGH profile margin. To realize the full quantum security margins stated in §3.5, EK and QK

© 2026 NQP LLC • All rights reserved • www.enqpy.com

MUST have min-entropy equal to the full key length (n nibbles = 4n bits). Keys generated directly from a CSPRNG meeting §3.4 requirements with a sufficient entropy source will satisfy the full entropy requirement. Keys derived via a KDF from lower-entropy input material should be evaluated against this advisory before HIGH profile deployment. Note on OR_EXP entropy ceiling: OR_EXP provides deterministic diffusion of OR_CTR across n key positions — not entropy expansion. The entropy of OR_EXP is bounded by the 64-bit OR_CTR regardless of n. The statistical uniqueness of eff_or is provided by the CSPRNG OR component. HIGH profile deployments MUST implement CSPRNG health monitoring (e.g., SP 800-90B continuous health tests) so that degradation of the CSPRNG cannot silently reduce the statistical uniqueness guarantee of eff_or.

3.5 Quantum Security Advisory

Profile	Quantum Search Cost	Advisory
LOW (128-bit)	2^{64} quantum ops	Below the NIST-recommended 2^{128} quantum security margin. Acceptable for non-critical, short-lived deployments where hardware constraints prohibit larger keys.
MEDIUM (192-bit)	2^{96} quantum ops	Provides substantial quantum security margin for near-term quantum capabilities.
HIGH (256-bit)	2^{128} quantum ops	Full NIST-recommended quantum security margin. Required for long-lived key relationships and high-value data protection.

Note: Quantum search cost figures in the table above assume full-entropy keys (min-entropy = key length). Keys at the n/2 minimum entropy floor provide half the stated quantum search cost in exponent — for example, HIGH profile at n/2-nibble entropy provides 2^{64} quantum search cost, not 2^{128}. See §3.4 Key Entropy Advisory.

Enqpy™'s underdetermination property is structurally independent of computational hardness and is not weakened by quantum computing. No quantum algorithm — including Grover's search or Shor's factoring algorithm — resolves the structural key ambiguity created by MOD16's many-to-one mapping. This is formally proved in [12], Corollary 2.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

4. Cipher Overview

4.1 System Model

Enqpy™ is a symmetric stream cipher. Two communicating parties share credentials pre-established out-of-band: OpenID (public), EK and QK (both secret), and an initial OR_CTR value of zero. For each message event the cipher executes five phases:

Phase	Name	Description
1	Initial Message Setup	Sender increments OR_CTR, generates a fresh CSPRNG OR, expands OR_CTR to n nibbles via PDAF (OR_EXP), and mixes OR with OR_EXP to produce eff_or. eff_or is public and transmitted with the ciphertext.
2	Key Generation	Derive per-session VKC and OKC from eff_or, EK, and QK via nonce-derived pointer keys (VKP, OKP) and PDAF Mode 1. Derive the per-session Case Selector (CS) via PDAF Mode 1.
3	Selection (W generation)	Generate the per-unit cipher key W via a CS-ordered, three-case PDAF-based traversal of VKC and OKC.
4	Cipher	Encrypt: W_byte ⊕PT_byte = CT_byte. Decrypt: W_byte ⊕ CT_byte = PT_byte.
5	Key Update	Derive VKNext and OKNext from current VKC/OKC via PDAF Mode 1 with cross-combined inputs. Executed when more plaintext or ciphertext remains after a full W cycle.

4.2 Operational Flow Summary

Send (Encrypt)

1. Increment OR_CTR (persist). Generate random OR. Compute:

```
OR_CTR_nibs ← 16-nibble representation of OR_CTR
OR_EXP[0..n-1] = PDAF(OR_CTR_nibs, OR_CTR_nibs, Mode=1, n_param=16)[0..n-1]
eff_or[i] = MOD16[or_nibs[i]][OR_EXP[i]]
```

2. VKP[0..n-1] = PDAF(eff_or, eff_or, Mode=1)[0..n-1] (nonce self-expansion)
3. OKP[i] = MOD16[VKP[i]][DS_SEP] where DS_SEP = 0xF
4. VKC = PDAF(EK, VKP, Mode=1); OKC = PDAF(QK, OKP, Mode=1)
5. CS = PDAF(VKC[0..n-1], OKC[0..n-1], Mode=1)[0..2]

```
case_order = CS_PERMUTATION_TABLE[CS mod 6]
```

6. Selection: CS-ordered three-case traversal → W nibble stream
7. Cipher: W_byte XOR PT_byte = CT_byte
8. Transmit [OpenID, eff_or, CT]

© 2026 NQP LLC • All rights reserved •

Receive (Decrypt)

9. Receive [OpenID, eff_or, CT]
10. Reproduce VKC, OKC, and CS identically using received eff_or and shared EK, QK.
11. Selection and Update: identical CS-ordered logic as sender.
12. Cipher: W_byte XOR CT_byte = PT_byte

© 2026 NQP LLC • All rights reserved • www.enqpy.com

5. Cryptographic Primitives

All security properties of Enqpy™ rest on two original NQP primitives: OWC (One-Way Computation) and PDAF (Pseudo-random Data Augmentation Function). These are not derived from existing cryptographic libraries. Both are constructed on modular-16 arithmetic, which provides a many-to-one mapping that is the fundamental source of the underdetermination property.

5.1 One-Way Computation (OWC)

5.1.1 Purpose and Role

OWC is used in the Nil-Communication Key Update mechanism (Section 7.4) to irreversibly derive a compact entropy seed from the shared key state. Its one-way property is structural, not computational.

5.1.2 Algorithm

OWC performs a digit-pair modular-16 combination pass over a nibble array. For each position i, the nibble at position i and the nibble at position i + nSkip are combined via MOD16. The position pointer advances by (1 + nSkip) per output nibble, producing non-overlapping pairs. If i + nSkip falls beyond the end of the array, the algorithm falls back to the adjacent nibble at position i + 1. The output is approximately half the length of the input.

5.1.3 One-Way Gate Property

The operation is not invertible: for any output nibble r, there exist exactly 16 input pairs (d1, d2) satisfying (d1 + d2) mod 16 = r. This is a structural many-to-one mapping — not computational hardness. For a key of n nibbles, the number of possible input keys consistent with any given OWC output of length m is at minimum 16^m.

5.1.4 Test Vector

Parameter	Value
Input (hex)	FCB578
nSkip	1
Calculation	Nibbles {15,12,11,5,7,8}: MOD16[15][12]=11, MOD16[11][5]=0, MOD16[7][8]=15
Output	B0F

5.2 Pseudo-random Data Augmentation Function (PDAF)

5.2.1 Purpose and Role

PDAF is the primary key expansion and key derivation primitive in Enqpy™. It is used in every phase that produces new key material: Phase 1 (OR_CTR expansion), Phase 2 (VKC and OKC derivation), Phase 2B (Case Selector derivation), Phase 3 (W generation), Phase 5 (in-cipher key update), and Nil-Communication Key Update.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

5.2.2 Mode 0 — Dual Key Pointer Add

Initialize pointer p = 0, cycle counter c = 0. For each output nibble: f1 ← ok[p % n]; f2 ← vk[(p + c) % n]; out[nN] ← MOD16_TABLE[f1][f2]. Increment p; when p ≥ n, reset p = 0 and increment c. Mode 0 is used as an internal component in the Nil-Communication Key Update construction (Section 7.4). It is not used in the primary cipher path.

5.2.3 Mode 1 — Key Offset Add

Both nibble arrays are tiled to minimum length 2n + 16 (sufficient for all per-phase calls where nDigits = n; the tiled arrays are indexed up to p + δ + 1 + c with maximum index 2n + 14). For each output nibble: f1 ← addKey[p + c]; δ ← pointKey[p]; f2 ← addKey[p + δ + 1 + c]; out[nN] ← MOD16_TABLE[f1][f2], where addKey is the tiled ValueKey array and pointKey is the tiled OffsetKey array. Mode 1 uses the OffsetKey nibble at each position as a dynamic displacement into the ValueKey.

The displacement structure creates a self-referential constraint system in the equations relating input to output: the OffsetKey controls which ValueKey positions are combined, but the OffsetKey is itself part of the unknowns when attempting inversion. This entanglement provides the Mode 1 underdetermination guarantee that is the foundation of Enqpy™'s key protection.

Tiling periodicity: the tiling of input arrays creates periodic structure in Mode 1 output. The n^2 output is not a uniformly random n^2-nibble string but has block structure with period n modulated by the cycle counter c. This means Mode 1 outputs form a proper subset of all n^2-nibble strings. The preimage lower bound $|P(O^*)| \geq 2$ (worst case) / ≥ 16 (non-degenerate) in [12] Theorem 1 is established for achievable outputs $O^* \in \text{image}(PDAF_1)$ — not for the set of all n^2-nibble strings. An adversary attempting inversion over Mode 1 outputs faces at least 2 consistent input keys (worst case) or at least 16 (non-degenerate) for any achievable output. The block structure does not reduce this floor; it restricts the domain of valid outputs over which the bound holds, which is the domain the proof characterizes.

Self-referential application (VK = OK): when PDAF Mode 1 is called with VK = OK (e.g., in Phase 1 OR_EXP derivation), the displacement at each output position is δ = VK_tiled[p], and both combined positions draw from the same tiled array. This is a distinct case from the two-independent-input construction and produces a self-referential constraint system upon inversion. The preimage lower bound of [12] Theorem 1 is established for the two-independent-input case; the self-referential case is used only in Phase 1 OR_EXP derivation, where the security requirement is one-way expansion of OR_CTR — a weaker property than general session key protection. The self-referential construction is the correct design for OR_EXP: there is no independent OffsetKey because OR_CTR is the sole input material.

5.2.4 One-Way Gate Property

PDAF Mode 1 is non-invertible due to the compounded structural underdetermination of its construction. The n^2 output is produced from n input nibbles (expansion ratio 1:n) via an OffsetKey-dependent displacement that creates entangled equations between the two input arrays. An adversary attempting inversion must simultaneously resolve both the ValueKey and the OffsetKey from the output — a task that faces genuine multi-solution ambiguity rather than the simple 16-solution reduction possible with Mode 0's linear chaining structure. The preimage lower bound is formally proved in [12], Theorem 1: $|P(O^*)| \geq 2$ (worst case: $\mu_odd \geq 1$) or ≥ 16 (non-degenerate: $\mu_odd = 0$) for any achievable output $O^* \in \text{image}(PDAF_1)$.

An adversary with full known-session-key material may use the tiling periodicity as a consistency filter, eliminating candidate keys whose Mode 1 outputs do not match the observed block structure. This filter reduces the search space from all n^2-nibble strings to Mode 1-consistent outputs, but does not reduce the preimage count below the proved minimum established in [12]. The filter narrows the search to the correct domain — n-nibble input keys — which is precisely what [12]

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Theorem 1 characterizes. Chosen-nonce cycle structure: the exact preimage count for a given output O* depends on the cycle structure of the PDAF Mode 1 dependency graph for the corresponding OK (see [12], §11 / Appendix B for the exact formula). For a known (VKC, OKC) pair, a chosen-nonce adversary controls eff_or and therefore indirectly influences the dependency graph structure. Whether a chosen-nonce adversary can force the dependency graph into a cycle structure that approaches the worst-case preimage floor (rather than the non-degenerate floor) is a theoretical question. The worst-case floor of $|P(O^*)| \geq 2$ per compatible OffsetKey (HIGH profile) remains unconditional; chosen-nonce attack can at most reduce the actual preimage count to this floor, not below it. This question is noted for completeness and does not affect the security proof bounds of [12].

This Mode 1 non-invertibility is the primary cryptographic barrier protecting the master keys EK and QK. The PDAF one-way gate between session key material (VKC, OKC) and master keys is the binding protection mechanism for EK and QK across all attack scenarios.

5.2.5 Randomness Property

When properly keyed with values from a CSPRNG meeting the requirements of Section 3.4, PDAF Mode 1 key streams pass statistical randomness testing. Empirical validation is presented in Section 14.

5.2.6 Test Vectors

Mode	ValueKey	OffsetKey	nDigits	Expected Output
Mode 0	FB382C001A	CC69100AB4	30	B7913C0ACE7FEBD00B53F4851014AF
Mode 1	FB382C001A	CC69100AB4	30	7DD02C010CDF74C01B5BF8D811B92B

© 2026 NQP LLC • All rights reserved • www.enqpy.com

6. PDAF_SEC Cipher — Complete Specification

6.1 Overview

PDAF_SEC is the Enqpy™ encrypt/decrypt function. It combines PDAF-based key derivation with an XOR cipher step applied at the byte level using packed W nibble pairs. The function is symmetric: the same operation with the same keys encrypts plaintext and decrypts ciphertext.

PDAF_SEC incorporates four architectural mechanisms that distinguish it from classical stream cipher constructions: PDAF-expanded OR_CTR nonce enforcement, domain-separated session key derivation, per-session key-dependent Selection case ordering via a PDAF Mode 1 Case Selector (CS), and cross-combined Phase 5 in-session key update.

6.2 Parameters

Parameter	Dir.	Description
ek	In	Encryption Key as nibble array. Length n must meet the active security profile minimum.
qk	In	Query Key as nibble array, same length n. QK ≠ EK required.
or_nibs	In	Open Return as nibble array, length n. CSPRNG-generated, never reused per (EK, QK) pair.
or_ctr	In	64-bit OR counter — monotonically increasing per credential pair.
n	In	Key length in nibbles. Must be ≥ 32 (LOW), ≥ 48 (MEDIUM), or ≥ 64 (HIGH) per active security profile.
target	In	Plaintext bytes (encrypt) or ciphertext bytes (decrypt).
nTextLen	In	Length of target in bytes.
out	Out	Caller-allocated output buffer, nTextLen bytes. Ciphertext length equals plaintext length exactly — zero overhead.
return	—	Number of bytes written, or −1 on error.

6.3 Phase Descriptions

Phase 1 — Initial Message Setup

The sender increments OR_CTR (persisted, pre-increment before use). A fresh n-nibble OR is generated from a CSPRNG. The OR_CTR is then expanded from its 64-bit integer form to n nibbles using PDAF Mode 1 self-expansion, providing strong diffusion of the counter across all key positions:

```
OR_CTR_nibs ← 16-nibble representation of the 64-bit OR_CTR value
OR_EXP[0..n-1] = PDAF(OR_CTR_nibs, OR_CTR_nibs, Mode=1, n_param=16)[0..n-1]
eff_or[i] = MOD16_TABLE[or_nibs[i]][OR_EXP[i]]
```

© 2026 NQP LLC • All rights reserved • www.enqpy.com

This construction ensures two independent uniqueness guarantees: the CSPRNG OR provides probabilistic uniqueness, and the OR_EXP component provides a hard deterministic uniqueness guarantee for all sequential OR_CTR values within the 2^{63} retirement threshold. The eff_or is transmitted in the clear alongside the ciphertext.

OR_EXP self-referential call: OR_EXP is derived via PDAF Mode 1 self-application of OR_CTR_nibs (VK = OK = OR_CTR_nibs). The preimage structure of this call is distinct from the primary cipher path (where VK ≠ OK). The deterministic uniqueness guarantee rests on OR_CTR monotonicity: no two messages with distinct sequential OR_CTR values under the same (EK, QK) credential pair, within the 2^{63} retirement threshold, produce identical OR_EXP. Note: the [+8] global shift property of $PDAF_1$ (see [12], Remark 6.1) means that any two OR_CTR values whose 16-nibble hex representations differ by exactly [8,8,...,8] mod 16 would produce identical OR_EXP; the first such pair in sequential use occurs at OR_CTR = 0x8888888888888888 ≈ 9.84×10^{18}, which exceeds the 2^{63} retirement threshold and is therefore unreachable in compliant deployments. The eff_or statistical uniqueness rests on CSPRNG OR independence. These are independent and complementary guarantees.

OR_EXP entropy scope: OR_EXP provides deterministic diffusion of OR_CTR across n key positions, not entropy expansion. The entropy of OR_EXP is bounded by the 64-bit OR_CTR: distinct sequential OR_CTR values within the 2^{63} retirement threshold always produce distinct OR_EXP values (deterministic injectivity), but OR_EXP cannot contain more than 64 bits of entropy regardless of n. The hard deterministic uniqueness guarantee means no two messages with distinct sequential OR_CTR values under the same credential pair will produce identical OR_EXP, up to OR_CTR = 2^{63}. The statistical uniqueness of eff_or is provided by the CSPRNG OR component. These are independent guarantees: deterministic (OR_CTR-based) and probabilistic (CSPRNG OR-based).

Phase 2 — Key Generation with Domain Separation

Using the received or reproduced eff_or, the cipher derives two session-specific working key pointers via nonce self-expansion and a fixed domain separator:

```
VKP[0..n-1] = PDAF(eff_or, eff_or, Mode=1)[0..n-1]  — nonce self-expansion (public, nonce-derived)
OKP[i] = (VKP[i] + DS_SEP) mod 16  where DS_SEP = 0xF  — domain-separated from VKP (public, nonce-derived)
VKC = PDAF(EK, VKP, Mode=1)
OKC = PDAF(QK, OKP, Mode=1)
```

The domain separator DS_SEP = 0xF guarantees structural independence between the VKP and OKP derivation paths, preventing trivial relationships such as VKP = OKP. It is a public constant and contributes no entropy. (Note: DS_VK = 0x5 and DS_OK = 0xA are the complementary domain separators used in EnqpyADS™ HMIX-based variant configurations; they are not used in the Ideal Configuration.) Because eff_or is unique per message, VKC and OKC are unique per message even if EK and QK remain unchanged across multiple messages.

Phase 2B — Case Selector Derivation

A per-session Case Selector (CS) is derived from the session working keys to vary the Selection case ordering. CS is derived using PDAF Mode 1, which provides the full entangled-displacement underdetermination property:

```
CS = PDAF(VKC[0..n-1], OKC[0..n-1], Mode=1)[0..2]
case_order = CS_PERMUTATION_TABLE[CS mod 6]
```

© 2026 NQP LLC • All rights reserved •

CS is a 3-nibble (12-bit) value. CS mod 6 selects one of the 3! = 6 possible orderings of Selection cases {1, 2, 3}: {1,2,3}, {1,3,2}, {2,1,3}, {2,3,1}, {3,1,2}, {3,2,1}. The selected ordering is applied cyclically throughout the Selection phase for that session.

> **CS DESIGN NOTE**
> CS mod 6 produces exactly 6 orderings — this is the intended design. The CS is not a work-factor amplifier; its role is equation-system separation (DOWN attack resistance: preventing reconstruction of the Selection equation system from known W values without session key knowledge) and per-session behavioral variation. The 12-bit CS derivation via PDAF Mode 1 ensures the active ordering is unknown to any adversary who has not recovered the session keys. The constant-time lookup of CS_PERMUTATION_TABLE (CS mod 6 → case ordering) is required to prevent timing side-channel leakage of the case ordering. Scope note: the 6-ordering space is a small, fixed enumeration. An adversary with known W values can enumerate all 6 orderings in negligible time; the security benefit of CS is the opacity of the active ordering before session key recovery — not exhaustion resistance. Descriptions of Enqpy™ MUST NOT represent CS as providing meaningful independent work factor against key-recovery attacks.

Phase 3 — Selection: W Generation

Selection applies the three cases in the CS-determined order, cycling pointer p = 0..n-1 and cycle counter C = 0..n-1.

Case	Expression	Description
Case 1 — Dual Key Add	W = MOD16[OKC[p]][VKC[p+C]]	OKC digit at p added to VKC digit at p+C.
Case 2 — VK Offset Add	W = MOD16[VKC[p+C]][VKC[p+δ+1+C]] where δ = OKC[p]	VKC digit at p+C added to VKC digit at offset position. δ derived from OKC.
Case 3 — OK Offset Add	W = MOD16[OKC[p+C]][OKC[p+δ_2 +1+C]] where δ_2 = VKC[p]	OKC digit at p+C added to OKC digit at offset position. δ_2 derived from VKC.

Single-key self-reference note: Cases 2 and 3 each draw both combined values from a single key array (VKC and OKC respectively), using the other array only as a displacement source. If the case ordering and the displacement-source array values are both known, Cases 2 and 3 produce single-array constraint equations of the form MOD16(K[a], K[b]) = W. Each such equation has exactly 16 consistent value pairs (the MOD16 fiber). A full-session known-plaintext attack over all n^2 W values yields at most 16^n jointly consistent key candidates per array — an upper bound from Lemma 1 (16 preimage pairs per nibble), consistent with the Mode 1 preimage lower bound proved in [12]. Complete decoupling of the VKC and OKC systems requires both the case ordering (known via 6-ordering exhaustive search) and knowledge of the displacement-source array values (which requires session key recovery), because OKC[p] appears as the displacement index in Case 2 equations, and VKC[p] appears as the displacement index in Case 3 equations.

Phase 4 — Cipher: XOR Encryption / Decryption

```
W_byte = (W_nibble_even << 4) | W_nibble_odd
CT_byte = W_byte XOR PT_byte  (encrypt)
PT_byte = W_byte XOR CT_byte  (decrypt)
```

© 2026 NQP LLC • All rights reserved • www.enqpy.com

The XOR is a single machine instruction — approximately 4 machine cycles per byte on any architecture. Ciphertext length equals plaintext length exactly; there is no expansion overhead.

Phase 5 — In-Session Key Update: VKNext and OKNext

After each complete traversal of the key space (n² W values per case), if more plaintext or ciphertext remains, both working keys are updated using Mode 1 PDAF with cross-combined inputs:

```
VKNext = PDAF(OKC[0..n-1], VKP, Mode=1)
OKNext = PDAF(VKC[0..n-1], OKP, Mode=1)
```

Phase 5 provides in-session key evolution that prevents reuse of W material across cycle boundaries within a session. An adversary who recovers the session keys (VKC, OKC) can advance the in-session key chain using VKP and OKP, which are deterministic functions of (EK, QK, eff_or). Cross-session isolation is provided exclusively by per-message eff_or uniqueness (OR_CTR + CSPRNG), not by Phase 5 key update.

CS re-derivation (REQUIRED): after updating VKC ← VKNext and OKC ← OKNext, CS MUST be re-derived: CS = PDAF(VKC[0..n-1], OKC[0..n-1], Mode=1)[0..2]; case_order = CS_PERMUTATION_TABLE[CS mod 6]. This re-derivation MUST be performed before beginning the next W generation cycle. Implementations that omit CS re-derivation will use a fixed case ordering across all update cycles, eliminating per-cycle case variation and reducing equation-system separation to one-time protection only.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

7. Key Management

7.1 Initial Credential Distribution

OpenID, EK, QK, and the initial OR_CTR value of zero are distributed out-of-band prior to first use. The initial key distribution channel must be assumed secure. Enqpy™ makes no claims about key establishment; it requires pre-shared secrets. EK and QK MUST be independently generated and MUST NOT be equal.

OpenID is a public routing identifier that specifies which (EK, QK) credential pair the receiver should use for decryption. OpenID is not a cryptographic identity claim and is not authenticated at the cipher layer in the standard construction. Substitution of OpenID by an adversary causes the receiver to attempt decryption with the wrong credential pair, which is detectable through MAC verification failure (if integrity is applied per §8.6) or plaintext structure validation failure. OpenID MUST uniquely identify the (EK, QK) credential pair within the receiver's credential set. OpenID need not be globally unique.

7.2 Per-Message OR Generation, Uniqueness Enforcement, and Credential Retirement

For each message, the sender: (1) increments OR_CTR by 1 and writes to non-volatile storage before the OR is generated; (2) generates a fresh n-nibble OR from a CSPRNG; (3) computes OR_EXP and eff_or as defined in Phase 1; (4) transmits eff_or alongside the ciphertext.

The OR_CTR component provides a hard uniqueness bound even if the CSPRNG produces a repeated OR value. The (EK, QK) credential pair MUST be retired when OR_CTR reaches 2^{63}.

OR_CTR PERSISTENCE REQUIREMENTS

1. OR_CTR write to non-volatile storage MUST be synchronous (fsync or hardware commit) before any encryption operation is considered committed for transmission. Write-back caching that defers the write until after transmission violates the deterministic uniqueness guarantee. 2. On restart following unclean shutdown (power loss, crash, watchdog reset), OR_CTR MUST be advanced by a guard increment of at least 1 before the next OR generation. Implementations MAY use a fixed guard increment of 2 to account for both the last unconfirmed increment and any write-back caching artifacts. 3. HIGH profile deployments MUST use hardware-secured monotonic counter storage (e.g., TPM NV counter, secure element) that guarantees atomic increment-then-commit semantics. OS-level persistence primitives alone do not satisfy this requirement. 4. The OR_CTR guard increment requirement applies regardless of OS-level persistence primitives and must be enforced at the cipher layer.

CREDENTIAL ROTATION GUIDANCE

OR_CTR Warning Threshold: When OR_CTR reaches 2^{62} (half of the retirement threshold), the sender SHOULD begin a credential rotation procedure. Rotation Options: (a) Nil-Communication Key Update (Method 1 or Method 2) may be used as a credential rotation mechanism. After a successful Nil-Comm update, OR_CTR is reset to 0 and the new (EK_new, QK_new) credential pair begins fresh. Both parties MUST agree on the rotation schedule before the threshold is reached. (b) Out-of-band re-keying with independently generated EK and QK is always acceptable and provides the strongest forward secrecy. Mandatory Retirement: The (EK, QK) credential pair MUST be retired and MUST NOT be used for any further encryption

© 2026 NQP LLC • All rights reserved • www.enqpy.com

when OR_CTR reaches 2^{63}. Implementations SHOULD enforce this limit at the cipher layer and return an error on any encryption attempt beyond this threshold.

7.3 In-Cipher Key Update (Phase 5)

The in-cipher key update evolves the session working keys automatically during encryption or decryption without any additional communication. The update occurs after each complete traversal of the key space (n^2 W values per case). Because the update uses Mode 1 PDAF with cross-combined inputs, the two key streams diverge after each update cycle. See Section 6.3 Phase 5 for the precise security boundary of this mechanism.

7.4 Nil-Communication Key Update

Enqpy™ supports complete evolution of the master keys EK and QK without any communication between participants. Both parties independently apply the same deterministic transformation to their shared key material, arriving at identical new key values. This eliminates the key-exchange attack surface that all other key management systems expose.

Method 1 — Deterministic Schedule Update

```
E_seed = OWC(PDAF(EK, QK, Mode=0)[0..n-1], nSkip=1)
EK_new = PDAF(EK, E_seed, Mode=1, n_param=n)[0..n-1]
QK_new = PDAF(QK, E_seed, Mode=1, n_param=n)[0..n-1]
OR_CTR = 0  (reset for new credential pair)
```

Note: E_seed has n/2 nibbles (OWC output); PDAF tiles E_seed cyclically to the required length.

Because E_seed is derived deterministically from the shared (EK, QK) state, both parties arrive at identical new keys with no communication.

METHOD 1 — TWO-LAYER PREIMAGE STRUCTURE AND FORWARD SECRECY

Method 1 uses a two-layer key derivation structure: (1) E_seed = OWC(PDAF(EK, QK, Mode=0)[0..n-1]) produces an n/2-nibble seed. The Mode 0 linear chaining structure provides a preimage space of 16^{n^2} at this stage. (2) EK_new = PDAF(EK, E_seed, Mode=1) applies a Mode 1 expansion. Recovery of EK from EK_new requires inverting the outer Mode 1 gate — which is the primary cryptographic barrier — and faces a preimage lower bound of ≥ 2 (worst case) or ≥ 16 (non-degenerate) per compatible OffsetKey, as proved in [12] Theorem 1, regardless of E_seed's entropy. FORWARD SECRECY: Method 1 is a deterministic forward chain. The key sequence from any update point forward is fully determined by the key state at that point. Compromise of any key state enables derivation of all future key states in that chain. Method 1 does not provide forward secrecy. For long-lived key relationships or HIGH profile deployments where forward secrecy is required, Method 2 MUST be used. Method 2 is the default for HIGH profile deployments; Method 1 requires explicit documented justification when used at HIGH profile.

Method 2 — External Entropy Injection

For HIGH profile deployments with long-lived key relationships, external entropy injection breaks the deterministic forward chain. Both parties incorporate a pre-agreed out-of-band entropy value (E_ext):

```
E_seed = OWC(PDAF(EK, QK, Mode=0)[0..n-1], nSkip=1)
```

© 2026 NQP LLC • All rights reserved • www.enqpy.com

```
E_combined[i] = MOD16[E_seed[i]][E_ext_nibs[i]]  (for i = 0..n/2-1)
EK_new = PDAF(EK, E_combined, Mode=1, n_param=n)[0..n-1]
QK_new = PDAF(QK, E_combined, Mode=1, n_param=n)[0..n-1]
```

Note: E_ext_nibs MUST have n/2 nibbles, matching E_seed length. E_seed and E_combined each have n/2 nibbles; PDAF tiles them cyclically to the required length.

E_ext ENTROPY REQUIREMENTS (MANDATORY)

E_ext MUST be generated by a CSPRNG meeting NIST SP 800-90A requirements, or be a sufficiently random out-of-band value from a documented entropy source. The min-entropy of E_ext MUST be ≥ n/2 nibbles (≥ 128 bits for HIGH profile), equal to the E_seed length. E_ext values derived from predictable or low-entropy sources — including timestamps, counters, device identifiers, sequence numbers, or any deterministic value — are prohibited and negate the forward secrecy property of Method 2. Each E_ext value MUST be used exactly once per key update event. Method 2 provides forward secrecy with respect to any adversary who cannot compromise future out-of-band E_ext values.

Method 2 is the required update method for HIGH profile deployments where key material protects data with a confidentiality requirement extending beyond the current session. Method 2 is the default Nil-Communication Key Update method for HIGH profile deployments; Method 1 MUST NOT be used as the default in HIGH profile implementations and MUST require explicit, documented justification when selected. Nil-Communication Key Update has no attack surface because there is no key exchange to intercept.

E_ext SYNCHRONIZATION PROTOCOL — REQUIRED BEFORE APPLYING METHOD 2

(1) Both parties independently compute E_seed = OWC(PDAF(EK, QK, Mode=0)[0..n-1], nSkip=1) from the current shared key state. No communication is required for this step. (2) Both parties compute the update commitment C = HMAC-SHA-256(E_seed, E_ext_bytes). Each party transmits C to the other through the out-of-band channel used to deliver E_ext. This commitment reveals neither E_ext nor EK; it confirms only that both parties hold the same E_ext for this update event. (3) Both parties verify that the received C matches their own computed C. If commitments do not match, the Method 2 update MUST NOT be applied. Both parties retain the current (EK, QK) state. The E_ext value is retired; a new E_ext must be established before retrying. (4) The key update MUST be applied at the same agreed message boundary: OR_CTR = N, where N is pre-agreed before the warning threshold. Both parties MUST apply the update after processing message N and before generating OR_CTR = N+1. (5) Recovery: if decryption fails after a scheduled Method 2 update, the receiver SHOULD attempt decryption with the pre-update key state as a single recovery attempt. If successful, the update was not applied by the sender; both parties SHOULD revert to the pre-update state and retry with a fresh E_ext. If recovery fails, escalate to out-of-band re-keying.

7.5 Key Size Flexibility

Enqpy™ supports any key size that meets or exceeds the active security profile minimum and is a multiple of one byte. Key size is fixed at credential establishment and MUST NOT be changed during operation of a given (EK, QK) credential pair. A key size change requires establishment of a new credential pair (new EK, QK with the target key size) through out-of-band key distribution or Nil-Communication Key Update to a new credential pair with the desired key length. In-operation dynamic key size change without credential renegotiation is not supported in this revision.

Note: the EnqpyADS™ BTA architecture supports key size variation across deployments and credential pairs. In-session key size transition protocol will be specified in a future revision.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

8. Security Analysis

8.1 Security Claims

Claim	Description
Shannon Ideal System	Enqpy™, in its Ideal Configuration, is the first finite-key cipher proved to satisfy Shannon's Ideal System definition. H(EK, QK | T^∞) = $\log_2$ (4) = 2 bits exactly (unconditional). The Ideal Configuration is the default implementation. EnqpyADS™ variant configurations carry per-session equivocation (Theorem 1) but not the formal global IS guarantee. Formally proved in [12]; see also §8.8 and §8.9.
Underdetermination	For any observed ciphertext CT, there exist at least two distinct plaintexts $PT_1 \neq PT_2$ and corresponding (EK_1 , QK_1 , OR_1) and (EK_2 , QK_2 , OR_2) such that PDAF_SEC(EK_1 , QK_1 , OR_1 , PT_1) = PDAF_SEC(EK_2 , QK_2 , OR_2 , PT_2) = CT. This is structural, not probabilistic. Preimage count formally bounded in [12].
Unconditional Quantum Resistance	The underdetermination property is platform-independent. A quantum computer cannot resolve the structurally absent information in the ciphertext. Algebraically proved in [12], Corollary 2.
One-Way Key Gates	Every application of OWC and PDAF Mode 1 places an irreversible gate between key versions. Full session key recovery does not expose master keys due to PDAF Mode 1 non-invertibility.
In-Session Key Evolution	Working keys for each cycle are derived from the prior session key state via Phase 5 PDAF Mode 1 update. This prevents reuse of W material across cycle boundaries within a session. Cross-session isolation is provided by per-message eff_or uniqueness (OR_CTR + CSPRNG), not by Phase 5 key update.
Plaintext Enqpy™, in its Ideal Configuration, achieves Equivocation H(PT|CT,OR) ≥ 128 bits (HIGH profile, n=64): for any ciphertext and public nonce, at least 2^128 plaintexts are consistent with (CT,OR), all equally probable. Independent of key equivocation. Proved in [12], Theorem 3. Immune to equivalent-key normalization.	Where the receiver can validate plaintext structure through application-layer means (e.g., protocol headers, format identifiers, checksums), correct decryption to a structurally valid result provides evidence of message origin. This is an application-layer inference, not a cryptographic authentication primitive. For arbitrary binary data, HMAC per Section 8.6 is MANDATORY.

© 2026 NQP LLC • All rights reserved •

Claim	Description
Structured-Plaintext Origin Inference	

MANDATORY INTEGRITY REQUIREMENT

The Structured-Plaintext Origin Inference claim applies only where the receiver can validate plaintext structure through application-layer means. For ANY deployment involving arbitrary binary data, compressed payloads, or encrypted inner layers where the receiver has no structural validation path, HMAC-SHA-256 per Section 8.6 is MANDATORY — not optional. Absence of a MAC exposes the deployment to ciphertext manipulation attacks.

8.2 PDAF Mode 1 as the Architectural Key Protection Boundary

The security of EK and QK depends on PDAF Mode 1 non-invertibility. HMIX is a nibble-wise MOD16 group operation: given HMIX output, eff_or, and the domain separator constant, the corresponding master key nibble is directly recoverable by subtraction mod 16. HMIX is therefore not a cryptographic barrier; it is a domain separation mechanism that guarantees structural independence (VKP ≠ OKP) and provides no secrecy.

The sole cryptographic barrier protecting EK and QK is the PDAF Mode 1 gate between the session keys (VKC, OKC) and the key pointers (VKP, OKP). Any reduction in the Mode 1 preimage count below its proved minimum directly reduces the master key security margin by the same factor. This architectural dependency is sound because PDAF Mode 1's entangled-displacement structure creates genuine underdetermination that is not reducible by any known method. For HIGH profile (n = 64), the preimage lower bound is ≥ 2 (worst case: μ_odd ≥ 1) or ≥ 16 (non-degenerate: μ_odd = 0) per compatible OffsetKey — unconditional and proved in [12], Theorem 1 and Corollary 2.

8.3 Perfect Security Cross Analysis

Attack Direction	Threat	Enqpy™ Defense
UP — Master Key Recovery	Recover EK/QK from a broken session.	PDAF Mode 1 one-way gate between VKC/OKC and VKP/OKP; HMIX inversion then provides direct QK/EK recovery — but VKP/OKP are unreachable through the PDAF Mode 1 gate. The PDAF Mode 1 gate is the sole cryptographic barrier; see Single-Barrier Architecture disclosure in §3.3.
RIGHT — Forward Propagation	Break future sessions from a broken session.	Per-message eff_or uniqueness (OR_CTR with PDAF expansion) ensures that each session uses unique key material. Phase 5 in-session key evolution does not bind future sessions to the plaintext; cross-session isolation is exclusively eff_or-based.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Attack Direction	Threat	Enqpy™ Defense
LEFT — Backward Propagation	Break past sessions from a broken session.	eff_or uniqueness and one-way PDAF update prevent backward propagation. Each session's VKC/OKC are unique functions of that session's eff_or.
DOWN — Partial KP Attack	Recover unknown PT from partial known-plaintext.	CS-ordered case selection makes the case ordering unknown without session key access. An adversary performing exhaustive 6-ordering search obtains the case partitioning of known W values. Within each case: Case 1 produces cross-array (OKC, VKC) constraints; Cases 2 and 3 produce single-array constraints (VKC-only and OKC-only) with displacement indices that depend on the other array. The effective work factor is: 6 ordering candidates × inversion of the resulting constraint system, where each system faces a preimage lower bound of ≥ 2 (worst case) or ≥ 16 (non-degenerate) per compatible OffsetKey over n-nibble session keys. Full decoupling of the VKC and OKC systems requires complete session key recovery.
LATERAL — Related-Nonce	Correlate keystreams from messages with related eff_or values.	Per-message eff_or uniqueness is the primary lateral attack defense. Domain separation constants DS_VK = 0x5 and DS_OK = 0xA are public structural constants guaranteeing VKP ≠ OKP (structural independence of the two derivation paths). They contribute no entropy and provide no secrecy.

8.4 Underdetermination Property — Formal Statement

For any ciphertext CT produced by Enqpy™ PDAF_SEC, there exist at least two distinct plaintexts $PT_1 \neq PT_2$ and corresponding credential and nonce combinations (EK_1, QK_1, OR_1) and (EK_2, QK_2, OR_2) such that PDAF_SEC(EK_1, QK_1, OR_1, PT_1) = PDAF_SEC(EK_2, QK_2, OR_2, PT_2) = CT. This property is inherent to the MOD16 structure at every key derivation, selection, and update step.

The exact preimage count is formally established in [12]: |P(O*)| = Σ_{compatible OK ∈ $\mathbb{Z}_{16}^n$} ∏_i f(μ_{i,odd}(OK)) [f(0)=16, f(k≥1)=2], where c(OK) is the number of connected components of the dependency graph and μ_{i,odd}(OK) is the number of independent odd-length cycles in component i. For HIGH profile (n = 64), the per-compatible-OK VK solution count is ≥ 2 (worst case: μ_odd≥1 per component; 2 solutions per component) and ≥ 16 (non-degenerate: μ_odd=0; 16 solutions per component), with the total scaling as 2^c (worst case) or 16^c (non-degenerate) across c connected components.

8.5 Known-Plaintext Properties

Enqpy™, as an Ideal System, is technically "susceptible" to known-plaintext in the following limited sense: if both a plaintext unit and its corresponding ciphertext unit are known, the W value for that

© 2026 NQP LLC • All rights reserved • www.enqpy.com

unit is revealed by W = PT ⊕CT. However, the Perfect Security Cross analysis ensures that revealing W for one unit does not reveal W for other units, past messages, future messages, or the master keys. Additionally, the CS-ordered case selection means that an adversary with known W values cannot reconstruct the Selection equation system without first recovering the session keys, because the active case ordering is unknown.

8.6 Integrity and Authentication Guidance

The Structured-Plaintext Origin Inference property applies where the receiver can validate plaintext structure through application-layer means. For arbitrary binary data, compressed payloads, or encrypted inner layers where the receiver has no structural validation path, applications MUST apply one of the following integrity mechanisms:

NORMATIVE INTEGRITY SPECIFICATION — RECOMMENDED METHOD

Integrity Key Derivation: Derive the HMAC key IK using HKDF-SHA-256 (RFC 5869) as follows: IK = HKDF-SHA-256(IKM=EK_bytes, salt=eff_or_bytes, info=0x456E717079494E54454752495459, [ASCII: EnqpyINTEGRITY] L=32) MAC Computation: Tag = HMAC-SHA-256(IK, eff_or_bytes || CT_bytes) Transmission: Transmit Tag alongside [OpenID, eff_or, CT]. Verification: Verify Tag using a constant-time comparison function before processing decrypted output. Return a generic error response regardless of whether MAC verification or PT structure validation fails, to prevent distinguishing oracles. IK lifecycle: IK is a per-message secret. IK MUST be zeroized from memory immediately after Tag computation (sender) or Tag verification (receiver). IK MUST NOT be stored, logged, or reused across messages. Security properties: (1) Implicit nonce binding — eff_or is authenticated both explicitly (as MAC input) and implicitly (through IK derivation). An adversary cannot replay a valid (eff_or, CT, Tag) triple under a substituted eff_or without access to EK, because IK would differ. (2) Scope-limited IK exposure — if IK for a given eff_or is exposed through a side channel, forgery is limited to (CT, Tag) pairs for that specific eff_or value only. Per-message IK re-derivation ensures other messages are unaffected.

HKDF info string registry: the info string 0x456E717079494E54454752495459 (EnqpyINTEGRITY) is the sole currently registered context label for Enqpy™ integrity key derivation. Any future extension of Enqpy™ that derives additional key material from EK using HKDF MUST use a distinct, registered info string. Reserved namespace: EnqpyINTEGRITY (current), EnqpyKEYCONFIRM (reserved), EnqpySESSIONID (reserved). New info strings MUST be registered with NQP LLC before deployment to prevent context collision.

Authenticated-sender-identity variant: for deployments where the receiver manages credentials for multiple senders and must prevent OpenID substitution from leaking credential-mapping information, include OpenID in the MAC input: Tag = HMAC-SHA-256(IK, OpenID_bytes || eff_or_bytes || CT_bytes). This variant authenticates OpenID cryptographically. The standard variant (without OpenID in MAC input) is suitable for single-credential-pair deployments.

Alternative: Structure plaintext to include a fixed-format header (e.g., a 32-byte random nonce followed by a structurally validated payload). Incorrect decryption will fail header validation at the application layer.

Minimum: Include a plaintext CRC or checksum within the plaintext structure. This provides error detection but not cryptographic authentication against an active adversary. AEAD construction roadmap: a future Enqpy™-AEAD construction will integrate authentication inseparably with encryption, eliminating the implementation risk of deploying PDAF_SEC without a MAC. Until

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Enqpy™-AEAD is available, the Recommended Method (HKDF + HMAC-SHA-256 per the normative box above) MUST be applied for all deployments handling arbitrary binary data, compressed payloads, or any content without reliable structure validation at the receiver. Implementations that expose PDAF_SEC encryption without co-requiring MAC application are non-conformant for binary data deployments.

8.7 Comparison with Standard Ciphers

Cipher	Security basis	Quantum	Key equivocation	Key size	Auth	Speed
Enqpy™ PDAF_SEC	Shannon underdetermination (Ideal System — proved [12,13])	Unconditional	≥ 1–4 bits (worst case / non-degenerate; c=1)	≥ 128–256 bit	Origin inference (structured PT); HMAC mandatory (binary data)	~2,900 MB/s (Phase 4 XOR; full PDAF_SEC ~399–454 MB/s — see §10.2)
AES-256-GCM	Computational hardness (block cipher)	Partial (Grover halves)	0 bits	256 bits fixed	AEAD tag 28 bytes	92–125 MB/s enc (software) 3,000+ MB/s (AES-NI)
ChaCha20-Poly1305	Computational hardness (ARX)	Partial (Grover halves)	0 bits	256 bits fixed	Poly1305 28 bytes	995–1,457 MB/s
NIST PQC (Kyber)	Lattice hardness (conditional)	Conditional	0 bits	Fixed param sets	None	Varies
One-Time Pad	Information-theoretic (Perfect Secrecy)	Unconditional	= msg len	= msg len	None	N/A

Note: AES-256-GCM software-only speed is shown for non-AES-NI environments. On x86 hardware with AES-NI and PCLMULQDQ instructions, AES-256-GCM achieves 3,000+ MB/s. Full benchmark environment documentation including AES-NI comparison, short-message latency figures, and per-payload overhead will be published in a forthcoming Enqpy™ Technical Supplement.

8.8 Formal Proof Status — First Provably Secure Shannon Ideal System

Enqpy™, in its Ideal Configuration, is the first finite-key cipher construction formally proved to satisfy Shannon's Ideal System definition — a specific information-theoretic property distinct from

© 2026 NQP LLC • All rights reserved • www.enqpy.com

computational security — with an exact equivocation floor of 2 bits. The proof is unconditional: it requires no computational hardness assumption and holds against any adversary, including quantum adversaries. Second Axis — Plaintext Equivocation (Theorem 3 in [12]): Independent of the key equivocation result, the companion Plaintext Equivocation Theorem establishes H(PT|CT,OR) ≥ 128 bits unconditionally for HIGH profile (n=64). For any ciphertext and public nonce, at least 2^128 plaintexts are consistent with the observation, all equally probable within the proved Case 1 achievable subset (Conjecture 1 in [12] extends uniformity to the full consistent set). This completes the Shannon Ideal System result on both axes: the adversary cannot determine either the key (2 bits of key equivocation) or the message (≥ 128 bits of plaintext equivocation). The plaintext-axis result is independent of equivalent-key normalization: the consistent plaintext set is a property of the cipher's keystream output, not of how keys are counted.

The One-Time Pad satisfies Perfect Secrecy (a strictly stronger property than the Ideal System) but requires a key equal in length to every message, rendering it impractical for general communications. Enqpy™ achieves the Ideal System property with a fixed, finite, reusable key.

Formal result (HIGH profile, n = 64 nibbles):

```
H(EK, QK | T^∞) = log₂(4) = 2 bits  (exact; Ideal Configuration)
H(EK, QK | C^m, M^m) ≥ 4 bits  (Layer 2 non-degenerate bound; non-degenerate: no odd-length cycles, c=1)
T_{>t} ⊥ (EK, QK) | T_{≤t}  (exact conditional independence; Ideal Configuration)  H(PT|CT,OR) ≥ 128 bits (HIGH profile, n=64) — message-axis result: at least 2^128 consistent plaintexts per ciphertext, all equally probable. Proved in companion document [12], Theorem 3.
```

The equivocation is exact and permanent: the ambiguity set is exactly the 4-element coset {EK, EK+8·1} × {QK, QK+8·1}. Structural source of the 2-bit exact floor: the irreducible 4-element ciphertext-equivalent coset is {EK, EK+8·1} × {QK, QK+8·1} — the four proved-secure operationally-indistinguishable key pairs related by nibble-wise addition of 8 (mod 16) to all positions of EK, QK, or both. All four pairs produce identical $PDAF_1$ outputs and identical ciphertext for all messages and all nonces, because adding 8 to both positions of any $PDAF_1$ ModSum produces a net shift of +16 ≡ 0 (mod 16), and in the Ideal Configuration neither EK nor QK appears as an OffsetKey parameter (so the [+8] shift propagates cleanly through both derivation paths). No observation of any transcript can distinguish the four coset elements. The 2-bit equivocation floor is the exact information-theoretic bound on adversarial key certainty. Configuration scope. This result holds for the Ideal Enqpy Configuration, in which EK and QK appear only as ValueKey parameters in their respective $PDAF_1$ calls, and VKP/OKP are derived from the public nonce alone. In the standard deployment profiles (LOW n=32, MED n=48, HIGH n=64), EK and QK also enter HMIX to derive VKP and OKP. At these key lengths (8 mod n ≠ 0), the [+8] shift on EK changes OKP, and $PDAF_1$ (QK, OKP+8) ≠ $PDAF_1$ (QK, OKP), breaking the coset. The standard profiles gain defense-in-depth key mixing but do not carry the provable global Ideal System guarantee. Deployments requiring the formal Shannon Ideal System property should use the Ideal Configuration.

Document	Content
[12] Formal Information-Theoretic Proof, Rev 1.0	Lemma 1 (MOD16 fiber size = 16, algebraic); Theorem 1 (PDAF Mode 1 preimage lower bound |P(O*)| ≥ 2 worst case / ≥ 16 non-degenerate); Theorem 2 (Shannon Ideal System — proved for Ideal Configuration: H(EK,QK|T^∞) = $\log_2$ (4) = 2 bits exact; T_{>t} ⊥ (EK, QK) | T_{≤t} conditional independence; six-step proof); Theorem 3 (plaintext equivocation: H(PT|CT,OR) ≥ 128 bits unconditionally for HIGH profile); Ideal Configuration definition and standard profile trade-off characterization (§6.1); Corollary 2 (quantum invariance). §11 (Appendix

Document	Content
	B) provides the extended algebraic proofs: exact preimage count formula |P(O*)| = Σ_{compatible OK} ∏_i f(μ_{i,odd}(OK)) [f(0)=16, f(k≥1)=2]; equivocation tightness (rank-ceiling argument over $\mathbb{Z}_{16}$); cycle case completeness; exact solution count with cycles. All proofs unconditional and constructive.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

8.9 Ideal Enqpy™ Configuration

8.9.1 Definition

The Ideal Enqpy™ Configuration is the specific key derivation arrangement under which the construction is formally proved to satisfy Shannon's Ideal System definition with an exact equivocation floor of $H(EK, QK \mid T^{\infty}) = \log_2(4) = 2$ bits. It is defined by a single structural principle: Key Role Separation Principle. Each master key (EK, QK) appears as the ValueKey parameter in exactly one PDAF Mode 1 call per session. Neither master key appears as an OffsetKey parameter or as input to HMIX for nonce pointer derivation.
Under this principle, the per-session key derivation proceeds as follows:

```
VKP_t  =  PDAF₁(OR_t, OR_t)[:n]          (nonce only — public)
OKP_t  =  (VKP_t + DS_SEP) mod 16        (nonce only — public, domain-
                                          separated)
VKC_t  =  PDAF₁(EK, VKP_t)               (EK as ValueKey only)
OKC_t  =  PDAF₁(QK, OKP_t)               (QK as ValueKey only)
W_t    =  CS_t-ordered Selection(VKC_t, OKC_t)
```

The OffsetKey parameters VKP_t and OKP_t are derived from the per-session public nonce OR_t alone, via the same PDAF Mode 1 self-referential nonce expansion already used in Phase 1 for OR_EXP (see Section 6.3). DS_SEP is a fixed domain separator constant distinguishing VKP from OKP.

8.9.2 Why This Configuration Delivers the Ideal System Property

The algebraic foundation is the [+8] global shift invariant of PDAF Mode 1: $PDAF_1(VK + 8 \cdot 1, OK) = PDAF_1(VK, OK)$ for all VK, OK $\in \mathbb{Z}_{16}^{n}$ and all n. When each master key appears only as ValueKey, this invariant propagates cleanly through both derivation paths:

- EK → EK + 8·1: VKP_t is derived from OR_t only (not EK), so VKP_t is unchanged. Therefore $VKC_t = PDAF_1(EK+8, VKP_t) = PDAF_1(EK, VKP_t) = VKC_t$ (unchanged). OKC_t is also unchanged, since EK does not enter its derivation.
- QK → QK + 8·1: Symmetrically, OKP_t is unchanged, so $OKC_t = PDAF_1(QK+8, OKP_t) = PDAF_1(QK, OKP_t) = OKC_t$ (unchanged). VKC_t is also unchanged.

Consequently, all four key pairs in the coset {EK, EK+8·1} × {QK, QK+8·1} produce identical VKC_t, OKC_t, W_t, and ciphertext C_t for every nonce OR_t and every plaintext. The coset is permanently and unconditionally transcript-indistinguishable. No accumulation of observed transcripts, however large, can separate the four coset elements. The exact equivocation floor $\log_2(4) = 2$ bits is therefore permanent and exact, not merely a lower bound.
The complete six-step proof of Theorem 2 (No-Later-Information, Version C — exact conditional independence) appears in [12] §6.1.

8.9.3 Operational Requirements

To operate Enqpy™ in the Ideal Configuration, the following requirements apply in addition to all standard operational requirements (Sections 3–7):

R1 — Nonce-Only Pointer Derivation

VKP and OKP MUST be derived from OR_t alone. Specifically:

- VKP_t MUST be computed as $PDAF_1(OR_t, OR_t)[:n]$ — the same self-referential nonce expansion used for OR_EXP in Phase 1, applied to the full nonce nibble representation and truncated to n nibbles.
- OKP_t MUST be derived from VKP_t by a fixed, publicly known domain separator addition: OKP_t[i] = (VKP_t[i] + DS_SEP) mod 16, where DS_SEP is a fixed constant (recommended: DS_SEP = 0xF, complementary to the OR_EXP domain structure). OKP MUST NOT incorporate EK, QK, or any other secret material.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

- The standard HMIX-based derivation (VKP = HMIX(QK, OR, DS_VK); OKP = HMIX(EK, OR, DS_OK)) MUST NOT be used for Ideal Configuration deployments.

R2 — Master Key Role Restriction

EK MUST appear only as the ValueKey parameter in $PDAF_1$ (EK, VKP_t). QK MUST appear only as the ValueKey parameter in $PDAF_1$ (QK, OKP_t). Neither EK nor QK may be introduced into HMIX calls, pointer derivation, or any OffsetKey position.

R3 — Phase 5 In-Session Key Update

The Phase 5 cross-combined key update (Section 6.3, Phase 5) is compatible with the Ideal Configuration because $PDAF_1$ is called with OKC_t and VKC_t as ValueKey parameters, and the [+8] invariant propagates through these calls. However, implementations MUST verify that VKP_t and OKP_t used in Phase 5 are the original nonce-derived values from R1, not values that incorporate master key material.

R4 — CS Derivation

The Case Selector CS_t = $PDAF_1$ (VKC_t, OKC_t)[0..2] is derived from the session working keys, not from EK or QK directly. This is unchanged from the standard construction and is compatible with the Ideal Configuration.

R5 — EK ≠ QK Requirement

The EK ≠ QK requirement (Section 3.3) remains mandatory. The Ideal Configuration preserves the two-key architecture: EK protects the VKC derivation path; QK protects the OKC derivation path. Setting EK = QK collapses the two-key separation and reduces the equivocation coset.

8.9.4 Relationship to Standard Deployment Profiles

EnqpyADS™ variant configurations may use alternative pointer derivation, such as HMIX-based derivation where EK enters HMIX to produce OKP and QK enters HMIX to produce VKP. This couples each master key into the OffsetKey path of the other's $PDAF_1$ call (defense-in-depth) at the cost of breaking the [+8] coset where 8 mod n ≠ 0. These are EnqpyADS™ variants, not alternative implementations of the Ideal Configuration.
The distinction between the Ideal Configuration and EnqpyADS™ variants is deliberate and explicit:

- Ideal Configuration (this document): formal Shannon Ideal System guarantee (H(EK,QK|T^∞) = 2 bits exactly); nonce-only pointer derivation; EK and QK as ValueKey only. Benchmarked performance is equivalent to HMIX-based variants across all message sizes.
- EnqpyADS™ HMIX-based variant: defense-in-depth key mixing; master keys protect the pointer derivation layer as well as the cipher output; per-session $PDAF_1$ preimage equivocation (Theorem 1) guaranteed; global Shannon IS property not formally proved. Appropriate for deployments where defense-in-depth takes precedence over the formal IS guarantee.

The Ideal Configuration is the default implementation specified in this document. Key Role Separation (EK and QK as ValueKey only; VKP and OKP derived from nonce alone) is the condition under which the exact coset-indistinguishability theorem holds for this $PDAF_1$ construction — the proof breaks if and only if a master key enters an OffsetKey path. Necessity and sufficiency within this construction are proved formally in [12] §6.1. The Ideal Configuration is therefore not a fragile or arbitrary wiring: it is precisely the minimal sufficient boundary.
Deployments with specific EnqpyADS™ requirements — defense-in-depth key mixing, hardware optimization, or other constraints — should consult the EnqpyADS™ specification for the appropriate variant configuration. There is no performance penalty for using the Ideal Configuration:

© 2026 NQP LLC • All rights reserved • www.enqpy.com

benchmark testing confirms equivalent throughput to HMIX-based variants across all message sizes (see §12 for benchmark results).

9. Singular Capabilities

Capability	Description
Shannon Ideal System — Proved	The only stream cipher other than the One-Time Pad proved to satisfy Shannon's Ideal System definition with a finite, reusable key — $H(EK,QK\|T^{\infty}) = \log_2(4) = 2$ bits exactly; $T_{>t} \perp (EK, QK) \mid T_{\leq t}$; unconditional, quantum-invariant. This is the Ideal Configuration — the default and sole proved-secure implementation in this document. EnqpyADS™ variants serve specific deployment requirements. Proof in [12].
Nil-Communication Key Update	Complete master key evolution between parties without any data exchange. Eliminates the key-exchange attack surface present in every other key management system. Method 1 provides deterministic key evolution (no forward secrecy). Method 2 (external entropy injection) provides forward secrecy and is required for HIGH profile long-lived key relationships. See Section 7.4.
Key-Dependent Selection Ordering	The Selection case ordering is derived from session keys VKC and OKC via PDAF Mode 1, making it unknown to any adversary without master key knowledge. CS mod 6 produces exactly 6 orderings by design, providing equation-system separation (DOWN attack resistance) and per-session behavioral variation.
Domain-Separated Nonce Mixing	VKP and OKP derivation use a fixed domain separator (DS_SEP = 0xF in the Ideal Configuration), ensuring structural independence of the two derivation paths (VKP ≠ OKP). This is a public constant and contributes no entropy. (EnqpyADS™ HMIX-based variants use DS_VK = 0x5 and DS_OK = 0xA.)
PDAF-Expanded Nonce Uniqueness	A 64-bit monotonic counter is expanded to n nibbles via PDAF Mode 1 before mixing with the CSPRNG-generated OR, providing strong diffusion across all nonce positions and a hard deterministic uniqueness guarantee.
Dynamic Variable Key Size	Key size is fixed per credential pair and defined at credential establishment. Flexible key size selection across deployments and credential pairs is supported via the EnqpyADS™ BTA architecture, subject to security profile minimums. See Section 7.5.
In-Session Key Evolution	Working keys for each cycle are derived from the prior session key state. Cross-session isolation is provided by per-message eff_or uniqueness.
Structured-PT Origin Inference	Correct decryption of structurally validated plaintext provides evidence of message origin. HMAC per Section 8.6 is mandatory for arbitrary binary data.
Single-Instruction Cipher Step	Phase 4 is a single XOR operation — approximately 4 machine cycles per byte on any architecture.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Capability	Description
< 4 KB Full Implementation	Complete cipher in under 4 KB of C source and under 270 lines of VHDL (Rev 1 sequential) / under 550 lines (Rev 3 parallel pipeline).
Platform Independence	Defined entirely in universal modular arithmetic. Produces identical output on any processor from 8-bit AVR to 64-bit server to FPGA.
Zero Ciphertext Overhead	Ciphertext length equals plaintext length exactly. The only transmitted overhead is eff_or (same length as the key) and an optional MAC tag.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

10. Performance and Efficiency

10.1 Architecture

Enqpy™'s cipher step (Phase 4) is a single XOR operation against a pre-computed W value. There are no rounds, no S-boxes, no Feistel networks, and no traditional key schedules. All computational cost is in the PDAF key expansion phases (Phases 1, 2, 2B, and 5), which operate on compact nibble arrays held entirely within L1 cache.

10.2 Speed

Algorithm	Enc (MB/s)	Dec (MB/s)	CT Overhead	Authentication
Enqpy™ PDAF_SEC	~2,900 MB/s (Phase 4 XOR; full PDAF_SEC ~399–454 MB/s — see §10.2)	~2,900 MB/s (Phase 4 XOR; full PDAF_SEC ~399–454 MB/s — see §10.2)	0 bytes	Structured-PT origin inference; HMAC mandatory (binary data)
AES-256-GCM	92–125 MB/s (sw) 3,000+ MB/s (AES-NI)	92–125 MB/s (sw) / 3,000+ MB/s (AES-NI)	28 bytes	AEAD tag
AES-256-CTR	650–922 MB/s	769+ MB/s	16 bytes	None
ChaCha20-Poly1305	995–1,457 MB/s	1,855+ MB/s	28 bytes	Poly1305 tag

AES-256-GCM software-only figures reflect non-AES-NI implementations. On modern x86 hardware with AES-NI and PCLMULQDQ instructions, AES-256-GCM achieves 3,000+ MB/s. The Enqpy™ ~2,900 MB/s figure reflects the Phase 4 XOR path with pre-computed W material amortized across large messages. A forthcoming Enqpy™ Technical Supplement will provide full benchmark environment documentation (CPU, compiler, flags), AES-NI comparison baselines, and per-message overhead at representative payload sizes (128 B, 1 KB, 64 KB, 1 MB).

In hardware FPGA implementations (VHDL Rev 3.0 parallel pipeline), Enqpy™ performs 256-bit encryption and decryption with approximately 15 clock cycles of key-setup latency, compared to approximately 4,310 cycles in the Rev 1 sequential implementation — a 287× improvement in key-setup latency. After key setup, Phase 4 streams bytes at 1 byte per clock cycle.

10.3 Code Size

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Implementation	Size	Notes
C reference (optimized)	< 4 KB (428 lines)	Complete cipher including all phases, lookup tables, PDAF OR_EXP, domain separation, and Case Selector derivation.
VHDL Rev 1.0 (sequential)	< 270 lines	Sequential FSM; one nibble per clock.
VHDL Rev 3.0 (parallel pipeline)	< 550 lines	Fully parallel pipeline; ~15 cycle key-setup latency for HIGH profile.

10.4 Bandwidth Overhead

The only values transmitted beyond the ciphertext are the OpenID (public identifier, not per-message overhead) and the eff_or (one per message, same length as the key). There is no authentication tag at the cipher layer, no nonce-mismatch overhead, and no protocol state machine required. Ciphertext length equals plaintext length exactly.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

11. Implementation Products

Product	Description
EnqpyChip™	Hardware implementation as FPGA/ASIC IP Core (Hard and Soft). Instantiates directly on any SoC. Based on VHDL Rev 3.0 parallel pipeline architecture. Provides on/off dynamic control interface.
EnqpySDK™	Software implementation as a cross-platform development kit. Compiled library in C/C++ and VHDL. Reference implementation: under 4 KB in C, under 550 lines in VHDL (Rev 3.0).

© 2026 NQP LLC • All rights reserved • www.enqpy.com

12. C Reference Implementation

12.1 Overview

*The Enqpy™ C reference implementation is available at github.com/nqp-llc/enqpy. The C reference is derived from the VHDL hardware reference and incorporates substantial implementation optimization. The optimization basis and the specific optimization techniques applied are documented separately in the **Enqpy™ Implementation Companion** (NQP LLC; available to commercial licensees under a separate NQP commercial agreement, to Partner Program participants, and to Foundation Conformance Program reviewers under a non-disclosure agreement).*

The optimized C implementation is verified correct against the naive (spec-literal) implementation by byte-exact output comparison across 128 MB of test data, and benchmarked to confirm throughput improvement under -O3 -march=native compilation.

12.2 Function Signatures (Section 12 — Complete API)

Initialization — Shared Lookup Tables

Call enqpy_init() once at program startup. This initializes the MOD16_TABLE lookup table. The persisted_or_ctr parameter is for caller documentation purposes; the caller manages OR_CTR persistence. EnqpySDK™ deployments MUST use the provided OR_CTR persistence wrapper (enqpy_or_ctr_init / enqpy_or_ctr_commit), which implements synchronous write, guard-increment-on-restart, and hardware monotonic counter integration for HIGH profile. Direct management of OR_CTR without this wrapper is permitted only for embedded targets that implement the full OR_CTR PERSISTENCE REQUIREMENTS of Section 7.2 at the application layer.

```
void enqpy_init(uint64_t persisted_or_ctr);
```

OWC — One-Way Computation

```
int OWC(const uint8_t *key_nibs, int nLen, int nSkip, uint8_t *out_nibs);
```

PDAF — Pseudo-random Data Augmentation Function

```
int PDAF(const uint8_t *vk, const uint8_t *ok, int n, int nMode, int
nDigits, uint8_t *out);
```

Case Selector Derivation

```
int ENQPY_DERIVE_CS(const uint8_t *vkc, const uint8_t *okc, int n, uint8_t
*cs_out);
```

ENQPY_DERIVE_CS uses a 3-position unrolled PDAF Mode 1 computation (c=0). The CS permutation lookup MUST be accessed in constant time to prevent timing side-channel leakage of the case ordering.

PDAF_SEC — Enqpy™ Encrypt / Decrypt

```
int PDAF_SEC(const uint8_t *ek, const uint8_t *qk, const uint8_t *or_nibs,
             uint64_t or_ctr, int n, const uint8_t *restrict target,
             int nTextLen, uint8_t *restrict out);
```

Nil-Communication Key Update

```
int ENQPY_NIL_COMM_UPDATE(const uint8_t *ek, const uint8_t *qk, int n,
```

© 2026 NQP LLC • All rights reserved • www.enqpy.com

```
                                const uint8_t *e_ext, int method,
                                uint8_t *ek_new, uint8_t *qk_new);
```

Set method=1 for deterministic schedule update, method=2 for external entropy injection. e_ext is ignored for method=1 and MUST be non-NULL and meet the entropy requirements of Section 7.4 for method=2.

12.3 Implementation Security Requirements

All buffers containing key material (EK, QK, VKC, OKC, VKP, OKP, W, OR_EXP) MUST be zeroed with memset_s() or equivalent secure erase before free(). OR_CTR MUST be written to non-volatile storage synchronously before use per Section 7.2. For HIGH profile deployments, hardware-secured monotonic counter storage is required. The CS permutation lookup (CS mod 6 → case ordering) MUST be implemented in constant time. Key material MUST NOT be logged, serialized, or passed to untrusted subsystems. Test vectors (Section 13) MUST be verified at initialization before processing live data. Applications using HMAC for integrity (Section 8.6) MUST compare MAC values using a constant-time comparison function to prevent timing oracle attacks. IK (the per-message HMAC key derived in §8.6) MUST be zeroized immediately after use.

EK ≠ QK enforcement: PDAF_SEC implementations SHOULD return −1 with an appropriate error code if EK and QK are byte-identical at the time of the call. Callers MUST verify EK ≠ QK at credential generation time and before each Nil-Communication Key Update. The cipher-layer check is a defense-in-depth measure; application-layer enforcement at credential generation time is the primary control.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

13. Test Vectors

13.1 OWC Test Vector

Parameter	Value
Input (hex)	FCB578
nSkip	1
Calculation	Nibbles {15,12,11,5,7,8}: MOD16[15][12]=11, MOD16[11][5]=0, MOD16[7][8]=15
Output	B0F

13.2 PDAF Test Vectors

Mode	ValueKey	OffsetKey	nDigits	Expected Output
0	FB382C001A	CC69100AB4	30	B7913C0ACE7FEBD00B53F4851014AF
1	FB382C001A	CC69100AB4	30	7DD02C010CDF74C01B5BF8D811B92B

13.2a PDAF Mode 1 Self-Referential Test Vector (VK = OK — Phase 1 OR_EXP)

The Phase 1 OR_EXP derivation calls PDAF Mode 1 with VK = OK = OR_CTR_nibs (the 16-nibble representation of the 64-bit OR_CTR value). This is a distinct case from the two-independent-input construction used in the primary cipher path. Implementations MUST verify OR_EXP output against the following vectors. The n_param=16 call produces the first 16 output nibbles used for truncation to n nibbles (HIGH profile n=64 uses all positions; LOW/MEDIUM profiles truncate to their respective n values from the same 64-nibble output).

OR_CTR (hex)	VK = OK (OR_CTR_nibs, 16 nibbles)	OR_EXP (16 nibbles, n_param=16)
0x0000000000000001	0000000000000001	0000000000000011
0x0000000000000002	0000000000000002	0000000000000022
0x000000000000A5C3	000000000000A5C3	00000000000AA5C3

Note: OR_CTR values 1 and 2 produce low-entropy OR_EXP output due to leading zeros in the 16-nibble representation. This is expected and correct — OR_EXP deterministic uniqueness rests on OR_CTR monotonicity, not on individual output entropy. Statistical uniqueness of eff_or is provided by the CSPRNG OR component mixed in Phase 1 (eff_or[i] = MOD16[or_nibs[i]][OR_EXP[i]]). OR_CTR 0xA5C3 shows the propagation behaviour for a non-trivial value.

13.3 PDAF_SEC Round-Trip Test

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Parameter	Value
EK	CB1E1203C479F30C1C356F12362FE43B47E8B5906C992013468395489A17D957
QK	0E2EAB25A9F78620ABB6726CF81A012776511B3988431D427DA911BDC2130680
OR (raw)	3667A507E1109EE32CD50718FA511065900EB422AC187AC5CD47EF5B18D86E0C
OR_CTR	0x0000000000000001 (first message)
DS_VK (EnqpyADS™ variant only)	0x5
DS_OK (EnqpyADS™ variant only)	0xA
Plaintext size	1,449,544 bytes
Ciphertext size	1,449,544 bytes (zero overhead confirmed)
Round-trip match	Byte-exact: all 1,449,544 bytes match original after decrypt(encrypt(PT))
Optimized vs. naive	Byte-exact output match confirmed across 128 MB test payload

The first 1,000,000 bits (125,000 bytes) of the resulting ciphertext were extracted and subjected to the complete NIST SP 800-22 test suite. See Section 14 for full results.

© 2026 NQP LLC • All rights reserved •

14. NIST SP 800-22 Randomness Test Results

14.1 Overview

The National Institute of Standards and Technology (NIST) Special Publication 800-22 provides 15 statistical tests designed to detect non-randomness in binary sequences. A sequence exhibits acceptable randomness properties if each test yields a p-value ≥ 0.01 (significance level α = 0.01).

14.2 Test Configuration

Parameter	Value
Sequence source	PDAF_SEC ciphertext output (first 1,000,000 bits)
Key length (n)	64 nibbles (256 bits per key)
EK	CB1E1203C479F30C1C356F12362FE43B47E8B5906C992013468395489A17D957
QK	0E2EAB25A9F78620ABB6726CF81A012776511B3988431D427DA911BDC2130680
OR	3667A507E1109EE32CD50718FA511065900EB422AC187AC5CD47EF5B18D86E0C
Plaintext source	TargetPlaintext.pdf (1,449,544 bytes, 26-page PDF document)
Sequence length	1,000,000 bits (125,000 bytes)
Significance level	α = 0.01
Round-trip check	Confirmed — byte-exact decryption verified prior to randomness testing

14.3 Test Results

#	Test Name	p-value	Result
1	Frequency (Monobit)	0.939419	**PASS**
2	Block Frequency (M=128)	0.636585	**PASS**
3	Runs	0.156777	**PASS**
4	Longest Run of Ones	0.771850	**PASS**
5	Binary Matrix Rank	0.325591	**PASS**
6	DFT (Spectral)	0.287107	**PASS**
7	Non-overlapping Template (m=9)	0.849404	**PASS**
8	Overlapping Template (m=9)	0.375944	**PASS**

© 2026 NQP LLC • All rights reserved • www.enqpy.com

#	Test Name	p-value	Result
9	Maurer's Universal Statistical	0.121548	**PASS**
10	Linear Complexity (M=500)	0.816379	**PASS**
11	Serial (m=16)	0.345325	**PASS**
12	Approximate Entropy (m=10)	0.323558	**PASS**
13	Cumulative Sums	0.316443	**PASS**
14	Random Excursions	0.211764	**PASS**
15	Random Excursions Variant	0.266222	**PASS**
		OVERALL RESULT:	**15 / 15 PASSED**

14.4 Interpretation

The p-values range from 0.121548 (Maurer's Universal) to 0.939419 (Frequency/Monobit), all well above the critical threshold of 0.01. No test exhibited values near the rejection boundary, indicating robust randomness properties across all statistical dimensions evaluated. These results are consistent with the theoretical underdetermination property of PDAF: the n^2 expansion from n input nibbles via modular-16 arithmetic, combined with the CS-ordered three-case Selection mechanism and per-cycle key updates, produces output that passes all standard tests for cryptographic randomness.

These results represent a single-sample test on one (EK, QK, OR) configuration and confirm that Enqpy™ PDAF_SEC output passes all NIST SP 800-22 Rev. 1a tests at the standard significance level of $\alpha = 0.01$. NIST SP 800-22 is a necessary but not sufficient condition for cryptographic quality. These results do not replace cryptographic security analysis. Multi-sample testing across independently generated key configurations, including related-key, related-nonce, all-zero-key, and all-ones-key samples, will be published in a forthcoming Enqpy™ Technical Supplement.

Acknowledgments

The author wishes to disclose that nonhuman artificial intelligence systems were utilized during the preparation of this document. Such use encompassed one or more of the following activities: information gathering, technical analysis, content creation, manuscript drafting, and editorial refinement. All cryptographic design decisions, security claims, intellectual contributions, and final determinations of technical accuracy remain solely those of the author. The underlying cryptographic system, its primitives, and its associated intellectual property are original works of the author.

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Appendix A — Enqpy™ Operational Flow Diagram

ENCRYPT

INITIAL SETUP
Credentials — OpenID, EK, QK, OR_CTR = 0 (pre-shared out-of-band)

PHASE 1 — Initial Message Setup:

```
Increment OR_CTR (write to non-volatile storage synchronously; hardware
counter for HIGH profile)
Generate random OR from CSPRNG (n nibbles)
OR_CTR_nibs ← 16-nibble representation of OR_CTR
OR_EXP[0..n-1] = PDAF(OR_CTR_nibs, OR_CTR_nibs, Mode=1, n_param=16)[0..n-
1]
eff_or[i] = MOD16_TABLE[or_nibs[i]][OR_EXP[i]]
```

PHASE 2 — Key Generation (Ideal Configuration):

```
VKP[0..n-1] = PDAF(eff_or, eff_or, Mode=1)[0..n-1]
OKP[i] = (VKP[i] + 0xF) mod 16
VKC = PDAF(EK, VKP, Mode=1)
OKC = PDAF(QK, OKP, Mode=1)
```

PHASE 2B — Case Selector Derivation (PDAF Mode 1, 3-nibble unrolled):

```
CS = PDAF(VKC[0..n-1], OKC[0..n-1], Mode=1)[0..2]
case_order = CS_PERMUTATION_TABLE[CS mod 6]   ← constant-time lookup
```

PHASE 3 — Selection (CS-ordered cases, cycling p = 0..n-1, C = 0..n-1):

```
Case 1: w = MOD16[OKC[p]][VKC[p+C]]
Case 2: w = MOD16[VKC[p+C]][VKC[p+d+1+C]]  where d = OKC[p]
Case 3: w = MOD16[OKC[p+C]][OKC[p+d2+1+C]]  where d2 = VKC[p]
```

PHASE 4 — Cipher (byte-level, nibble-packed):

```
W_byte = (W_nibble_even << 4) | W_nibble_odd
CT_byte = W_byte XOR PT_byte
```

PHASE 5 — In-Session Key Update (when cycle exhausted and more PT remains):

```
VKNext = PDAF(OKC[0..n-1], VKP, Mode=1)
OKNext = PDAF(VKC[0..n-1], OKP, Mode=1)
VKC ← VKNext, OKC ← OKNext; re-derive CS
```

SEND: [OpenID, eff_or, CT] to recipient

© 2026 NQP LLC • All rights reserved • www.enqpy.com

DECRYPT

RECEIVE: [OpenID, eff_or, CT]

NOTE: The receiver does not execute Phase 1. eff_or is received directly from the sender and used as-is. PHASE 2: Reproduce VKC and OKC identically using received eff_or and shared EK, QK.

PHASE 2B: Reproduce CS and case_order identically.

PHASE 3 + 4 + 5: Identical CS-ordered pointer and nibble-packing logic as sender.

```
PT_byte = W_byte XOR CT_byte  (XOR is self-inverse)
```

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Appendix B — Extended Test Vector Reference

B.1 PDAF Mode 0 Full n² Output (n = 16, 256 nibbles)

ValueKey: 0123456789ABCDEF | OffsetKey: FEDCBA9876543210

```
FFFFFFFFFFFFFFFF0000000000000000111111111111111122222222222222223333333333333333444444444444444455555555555555556666666666666666777777777777777788888888888888889999999999999999AAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBCCCCCCCCCCCCCCCCDDDDDDDDDDDDDDDDEEEEEEEEEEEEEEEE
```

B.2 PDAF Mode 1 Full n² Output (n = 16, 256 nibbles)

ValueKey: 0123456789ABCDEF | OffsetKey: FEDCBA9876543210

```
0123456789ABCDEF23456789ABCDEF01456789ABCDEF01236789ABCDEF01234589ABCDEF01234567ABCDEF0123456789CDEF0123456789ABEF0123456789ABCD0123456789ABCDEF23456789ABCDEF01456789ABCDEF01236789ABCDEF01234589ABCDEF01234567ABCDEF0123456789CDEF0123456789ABEF0123456789ABCD
```

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Appendix C — Security Definitions

C.1 Shannon Perfect Secrecy (OTP Form)

A cryptosystem achieves Perfect Secrecy if, for all messages M and all ciphertexts C: Pr[M = m | C = c] = Pr[M = m] for all m, c. Observing the ciphertext provides no information about the plaintext. Shannon proved this requires |K| ≥ |M|.

C.2 Shannon Ideal System

A cryptosystem is an Ideal System if the equivocation $H(K \mid C^n, M^n)$ does not approach zero as $n \to \infty$. In an Ideal System, even with unlimited ciphertext, the adversary is left with multiple alternative keys and plaintexts of reasonable probability. Enqpy™ is the first finite-key cipher formally proved to satisfy this definition; see [12].

C.3 Underdetermination

A cryptosystem is underdetermined if, for every ciphertext C, there exist multiple (PT, key) pairs that produce C. Enqpy™ achieves underdetermination at every step of key derivation, selection, and update via the many-to-one property of MOD16 addition.

C.4 PDAF Mode 1 Entangled Underdetermination

PDAF Mode 1 creates a stronger form of underdetermination than a simple many-to-one mapping. The OffsetKey controls dynamic displacements into the ValueKey during output generation, creating an entangled equation system when attempting inversion: the displacements that determine which ValueKey positions are combined are themselves the unknowns being sought. This self-referential constraint structure prevents the equation system from reducing to a small fixed number of solutions. The exact solution count is characterized in [12], Theorem 1 and §11 (Appendix B).

C.5 Computational Security (for comparison)

A cipher is computationally secure if breaking it requires resources exceeding a computational bound. AES, ChaCha20, and all NIST PQC standards are computationally secure but not unconditionally secure. Enqpy™ claims unconditional security through structural underdetermination, formally proved in [12].

© 2026 NQP LLC • All rights reserved • www.enqpy.com

Appendix D — References

[1] Shannon, C. E. (1949). Communication Theory of Secrecy Systems. Bell System Technical Journal, 28(4), 656–715.

[2] McGough, P. (2010, updated 2020). Qwyit Protocol Reference, V3.0. Qwyit LLC [now NQP LLC].

[3] NQP LLC (2021). QwyitCipher (Enqpy™) PDAF_SEC Cipher Configuration — Reference Guide V2.0.

[4] NQP LLC (2026). Enqpy™ Stream Cipher — Technical Description, Features, Benefits, and Singular Capabilities.

[5] NQP LLC (2026). Enqpy™ Flexible Security — Statement of BTA Capabilities V1.

[6] NIST (2010). SP 800-22 Rev. 1a — A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications.

[7] NIST (2012). SP 800-90A — Recommendation for Random Number Generation Using Deterministic Random Bit Generators.

[8] NIST (2018). SP 800-90B — Recommendation for the Entropy Sources Used for Random Bit Generation.

[9] NIST (2024). Post-Quantum Cryptography Standardization. FIPS 203 (ML-KEM), FIPS 204 (ML-DSA), FIPS 205 (SLH-DSA).

[10] Bernstein, D. J. (2008). ChaCha, a variant of Salsa20. Workshop Record of SASC.

[11] NIST (2001). Advanced Encryption Standard (AES). FIPS PUB 197.

Formal Proof Documents

[12] McGough, P. / NQP LLC (2026). Enqpy™ Stream Cipher: Constructive Proof of Shannon's Ideal System for a Finite-Key Cipher, Revision 1.0. IACR ePrint Archive, 2026. Establishes Lemma 1 (MOD16 fiber size), Theorem 1 (PDAF Mode 1 preimage lower bound |P(O*)| ≥ 2 worst case / ≥ 16 non-degenerate), Theorem 2 (Shannon Ideal System — proved for Ideal Enqpy Configuration: H(EK,QK|T^∞) = $\log_2$ (4) = 2 bits exact; conditional independence T_{>t} ⊥ S_t | T_{≤t}; six-step proof), Theorem 3 (plaintext equivocation: H(PT|CT,OR) ≥ 128 bits unconditionally for HIGH profile, with posterior uniformity on the Case 1 achievable subset and Conjecture 1 extending uniformity to the full consistent set), Corollary 2 (quantum invariance), and the Ideal Configuration definition with standard-profile trade-off characterization. §11 (Appendix B) provides the extended algebraic proofs: exact preimage count formula |P(O*)| = Σ_{compatible OK} ∏_i f(μ_{i,odd}(OK)) [f(0)=16, f(k≥1)=2]; equivocation tightness (rank argument over $\mathbb{Z}_{16}$); cycle case completeness ($\mathbb{Z}_{16}$ vs. $\mathbb{Z}_2$); exact solution count with cycles (induction proof). All proofs are unconditional and constructive.

© 2026 NQP LLC • All rights reserved • www.enqpy.com